Received: 28 July 2016 | Revised: 25 April 2017 | Accepted: 8 May 2017 | Published on: 13 June 2017

DOI: 10.1002/sam.11348

ORIGINAL ARTICLE

WILEY

Random forest missing data algorithms

Fei Tang | **Hemant Ishwaran**

Division of Biostatistics, University of Miami Coral Gables, Florida

Correspondence
Hemant Ishwaran, Division of Biostatistics, University of Miami, 1320 S Dixie Hwy, Coral Gables, FL 33146. Email: hemant.ishwaran@gmail.com

Funding information
National Institutes of Health, R01CA163739.

Random forest (RF) missing data algorithms are an attractive approach for imputing missing data. They have the desirable properties of being able to handle mixed types of missing data, they are adaptive to interactions and nonlinearity, and they have the potential to scale to big data settings. Currently there are many different RF imputation algorithms, but relatively little guidance about their efficacy. Using a large, diverse collection of data sets, imputation performance of various RF algorithms was assessed under different missing data mechanisms. Algorithms included proximity imputation, on the fly imputation, and imputation utilizing multivariate unsupervised and supervised splitting—the latter class representing a generalization of a new promising imputation algorithm called missForest. Our findings reveal RF imputation to be generally robust with performance improving with increasing correlation. Performance was good under moderate to high missingness, and even (in certain cases) when data was missing not at random.

KEYWORDS

correlation, imputation, machine learning, missingness, splitting (random, univariate, multivariate, unsupervised)

1 | INTRODUCTION

Missing data is a real-world problem often encountered in scientific settings. Missing data are problematic as many statistical analyses require complete data. This forces researchers who want to use a statistical analysis that requires complete data to choose between imputing data or discarding missing values. But to simply discard missing data is not a reasonable practice, as valuable information may be lost and inferential power compromised [7]. Thus, imputing missing data in such settings is a more reasonable and practical way to proceed.

While many statistical methods have been developed for imputing missing data, many of these perform poorly in high dimensional and large-scale data settings; for example, in genomic, proteomic, neuroimaging, and other high-throughput problems. In particular, it is generally recommended that all variables be included in multiple imputation to make it proper in general and in order not to create bias in the estimate of the correlations [18]. But this can lead to overparameterization when there are a large number of variables and the sample size is moderate. Computational issues may also arise. An example is the occurrence of non-convexity due to missing data when maximizing the log-likelihood. This creates problems for traditional optimization methods such as the EM (expectation–maximization) algorithm [16]. Missing data methods are also often designed only for continuous data (eg, gene expression data [1]), and for methods applicable to mixed data (ie, data having both nominal and categorical variables), implementation can often break down in challenging data settings [12]. Another concern is the inability to deal with complex interactions and nonlinearity of variables. Standard multiple imputation approaches do not automatically incorporate interaction effects, which leads to biased parameter estimates when interactions are present [6]. Although some techniques, such as fully conditional specification of covariates can be tried to resolve this problem [2], these techniques can be difficult and inefficient to implement in settings where interactions are expected to be complicated.

For these reasons there has been much interest in using machine learning methods for missing data imputation. A promising approach can be based on Breiman's random forests [3] (abbreviated hereafter as RF). RF have the desired characteristic that they: (1) handle mixed types of missing data; (2) address interactions and nonlinearity; (3)

 © 2017 Wiley Periodicals, Inc.

19321872, 2017, 6, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/sam.11348 by Technical University Dortmund, Wiley Online Library on [22/02/2023]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

scale to high-dimensions while avoiding overfitting; and (4) yield measures of variable importance useful for variable selection. Currently there are several different RF missing data algorithms. This includes the original RF proximity algorithm proposed by Breiman [4] implemented in the `randomForest` R-package [13]. A different class of algorithms are the "on-the-fly-imputation" (OTFI) algorithms implemented in the `randomSurvivalForest` R-package [11], which allow data to be imputed while simultaneously growing a survival tree. These algorithms have been unified within the `randomForestSRC` R-package (abbreviated as RF-SRC) to include not only survival but also classification and regression among other settings [10]. A third approach is missForest, a method recently introduced in [21]. Missforest takes a different approach by recasting the missing data problem as a prediction problem. Data are imputed by regressing each variable in turn against all other variables and then predicting missing data for the dependent variable using the fitted forest. MissForest has been shown [26] to outperform well-known methods such as k-nearest neighbors (KNN) [22] and parametric MICE [25] (multivariate imputation using chained equation).

Given that RF meets all the characteristics for handling missing data, it seems desirable to use RF for imputing data. However, practitioners have very little guidance about which of these algorithms to use, as there have been no systematic studies looking at the comparative behavior of RF missing data algorithms. Comparisons of RF imputation to other procedures have been considered by [21,26], and there have been studies looking at effectiveness of RF imputation when combined with other methods (for instance Shah et al. [20] showed that parameter estimates were less biased when using MICE with RF-based imputation), but no systematic study of performance between RF algorithms has been attempted. To address this gap in the literature, we therefore sought to study various RF algorithms and systematically document their performance using a large empirical study involving 60 data sets. Performance was assessed by imputation accuracy and computational speed. Different missing data mechanisms (missing at random [MAR] and not missing at random [NMAR]) were used to assess robustness. In addition to the RF missing data algorithms described above, we also considered several new algorithms, including a multivariate version of missForest, referred to as mForest. Despite the superior performance of missForest (a finding confirmed in our experiments), the algorithm is computationally expensive to implement in high-dimensions as a separate forest must be calculated for each variable and the algorithm is run until convergence is achieved. The mForest algorithm helps to alleviate this problem by grouping variables and using multivariate forests with each group used in turn as the set of dependent variables. This replaces p regressions, where p is the number of variables, with $\approx 1/\alpha$ regressions, where $0 < \alpha < 1$ is a user-specified group fraction size. Computational savings were found to be substantial for mForest without overly compromising accuracy even for relatively large α. Other studied RF algorithms included a new multivariate unsupervised algorithm and algorithms utilizing random splitting.

All forests constructed in the manuscript followed the RF methodology of [3]. Trees were grown using independently sampled bootstrap data. For univariate regression, continuous variables were split using squared-error splitting and categorical variables by the Gini index [5]. More general splitting rules, such as unsupervised and multivariate splitting, were also employed, and are described later in the manuscript. Random feature selection was used, with `mtry` variables selected at random prior to splitting a tree node, and trees grown as deeply as possible subject to the constraint of a lower bound of `nodesize` unique data values within a terminal node. RF missing data algorithms were implemented using the `randomForestSRC` R-package [10], which has been extended to include a new `impute.rfsrc` function optimized specifically for data imputation. The package implements openMP parallel processing, which allows for parallel processing on user desktops as well as large-scale computing clusters; thus greatly reducing computational times.

2 | RF APPROACHES TO IMPUTATION

Three general strategies have been used for RF missing data imputation:

1. Preimpute the data; grow the forest; update the original missing values using proximity of the data. Iterate for improved results.
2. Simultaneously impute data while growing the forest; iterate for improved results.
3. Preimpute the data; grow a forest using in turn each variable that has missing values; predict the missing values using the grown forest. Iterate for improved results.

Proximity imputation [4] uses strategy A, OTFI [11] uses strategy B, and missforest [21] uses strategy C. The way these algorithms impute data is very different. Even the manner in which they model outcome data (ie "Y" variables) is different. Strategy C does not use outcome data at all and is solely a technique for imputing features ($\mathbf{X}$ variables), while strategies A and B utilize outcome information in their model building. Below we detail each of these strategies and describe various algorithms which utilize these approaches. These algorithms take advantage of new splitting rules, including random splitting, unsupervised splitting, and multivariate splitting [10].

2.1 | Strawman imputation

We first begin by describing a "strawman imputation" which will be used throughout as our baseline reference value. While

19321872, 2017, 6, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/sam.11348 by Technical University Dortmund, Wiley Online Library on [22/02/2023]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

this algorithm is rough, it is very rapid, and for this reason it was also used to initialize some of our RF procedures. Strawman imputation is defined as follows. Missing values for continuous variables are imputed using the median of non-missing values, and for missing categorical variables, the most frequently occurring non-missing value is used (ties are broken at random).

2.2 | Proximity imputation: RF_{prx} and RF_{prxR}

Here we describe proximity imputation (strategy A). In this procedure the data are first roughly imputed using strawman imputation. A RF is fit using this imputed data. Using the resulting forest, the $n \times n$ symmetric proximity matrix (n equals the sample size) is determined where the (i,j) entry records the inbag frequency that cases i and j share the same terminal node. The proximity matrix is used to impute the original missing values. For continuous variables, the proximity weighted average of non-missing data is used; for categorical variables, the largest average proximity over non-missing data is used. The updated data are used to grow a new RF, and the procedure is iterated.

We use RF_{prx} to refer to proximity imputation as described above. However, when implementing RF_{prx} we use a slightly modified version that makes use of random splitting in order to increase computational speed. In random splitting, `nsplit`, a nonzero positive integer, is specified by the user. A maximum of `nspit`-split points are chosen randomly for each of the randomly selected `mtry` splitting variables. This is in contrast to nonrandom (deterministic) splitting typically used by RF, where all possible split points for each of the potential `mtry` splitting variables are considered. The splitting rule is applied to the *nsplit* randomly selected split points and the tree node is split on the variable with random split point yielding the best value, as measured by the splitting criterion. Random splitting evaluates the splitting rule over a much smaller number of split points and is therefore considerably faster than deterministic splitting.

The limiting case of random splitting is pure random splitting. The tree node is split by selecting a variable and the split-point completely at random—no splitting rule is applied; that is, splitting is completely nonadaptive to the data. Pure random splitting is generally the fastest type of random splitting. We also apply RF_{prx} using pure random splitting; this algorithm is denoted by RF_{prxR}.

We implement iterated versions of RF_{prx} and RF_{prxR}. To distinguish between the different algorithms, we write $RF_{prx}^{(k)}$ and $RF_{prxR}^{(k)}$ when they are iterated $k \geq 1$ times. Thus, $RF_{prx}^{(5)}$ and $RF_{prxR}^{(5)}$ indicate that the algorithms were iterated 5 times, while $RF_{prx}^{(1)}$ and $RF_{prxR}^{(1)}$ indicate that the algorithms were not iterated. However, as this latter notation is somewhat cumbersome, for notational simplicity we will simply use RF_{prx} to denote $RF_{prx}^{(1)}$ and RF_{prxR} for $RF_{prxR}^{(1)}$.

2.3 | On-the-fly-imputation: RF_{otf} and RF_{otfR}

A disadvantage of the proximity approach is that OOB (out-of-bag) estimates for prediction error are biased [4]. Further, because prediction error is biased, so are other measures based on it, such as variable importance. The method is also awkward to implement on test data with missing values. The OTFI method [11] (strategy B) was devised to address these issues. Specific details of OTFI can be found in [10,11], but for convenience we summarize the key aspects of OTFI below:

1. Only non-missing data are used to calculate the split-statistic for splitting a tree node.
2. When assigning left and right daughter node membership if the variable used to split the node has missing data, missing data for that variable are "imputed" by drawing a random value from the inbag non-missing data.
3. Following a node split, imputed data are reset to missing and the process is repeated until terminal nodes are reached. Note that after terminal node assignment, imputed data are reset back to missing, just as was done for all nodes.
4. Missing data in terminal nodes are then imputed using OOB non-missing terminal node data from all the trees. For integer-valued variables, a maximal class rule is used and a mean rule is used for continuous variables.

It should be emphasized that the purpose of the "imputed data" in Step 2 is only to make it possible to assign cases to daughter nodes—imputed data are not used to calculate the split-statistic, and imputed data are only temporary and reset to missing after node assignment. Thus, at the completion of growing the forest, the resulting forest contains missing values in its terminal nodes and no imputation has been done up to this point. Step 4 is added as a means for imputing the data, but this step could be skipped if the goal is to use the forest in analysis situations. In particular, step 4 is not required if the goal is to use the forest for prediction. This applies even when test data used for prediction have missing values. In such a scenario, test data assignment works in the same way as in step 2. That is, for missing test values, values are imputed as in step 2 using the original grow distribution from the training forest, and the test case assigned to its daughter node. Following this, the missing test data are reset back to missing as in step 3, and the process is repeated.

This method of assigning cases with missing data, which is well suited for forests, is in contrast to surrogate splitting utilized by CART [5]. To assign a case having a missing value for the variable used to split a node, CART uses the best surrogate split among those variables not missing for the case. This ensures every case can be classified optimally, whether the case has missing values or not. However, while surrogate splitting works well for CART, the method is not well suited for forests. Computational burden is one issue. Finding

19321872, 2017, 6, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/sam.11348 by Technical University Dortmund, Wiley Online Library on [22/02/2023]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

a surrogate split is computationally expensive even for 1 tree, let alone for a large number of trees. Another concern is that surrogate splitting works tangentially to random feature selection used by forests. In RF, candidate variables for splitting a node are selected randomly, and as such the candidate variables may be uncorrelated with one another. This can make it difficult to find a good surrogate split when missing values are encountered, if surrogate splitting is restricted to candidate variables. Another concern is that surrogate splitting alters the interpretation of a variable, which affects measures such as variable importance measures [11].

To denote the OTFI missing data algorithm, we will use the abbreviation RF_{otf}. As in proximity imputation, to increase computational speed, RF_{otf} is implemented using `nsplit` random splitting. We also consider OTFI under pure random splitting and denote this algorithm by RF_{otfR}. Both algorithms are iterated in our studies. RF_{otf}, RF_{otfR} will be used to denote a single iteration, while $\text{RF}_{\text{otf}}^{(5)}$ and $\text{RF}_{\text{otfR}}^{(5)}$ denote 5 iterations. Note that when OTFI algorithms are iterated, the terminal node imputation executed in step 4 uses inbag data and not OOB data after the first cycle. This is because after the first cycle of the algorithm, no coherent OOB sample exists.

Remark 1. As noted by one of our referees, missingness incorporated in attributes (MIA) is another tree splitting method which bypasses the need to impute data [24,23]. Again, this only applies if the user is interested in a forest analysis. MIA accomplishes this by treating missing values as a category which is incorporated into the splitting rule. Let X be an ordered or numeric feature being used to split a node. The MIA splitting rule searches over all possible split values s of X of the following form:

Split A: $\{X \le s \text{ or } X = \text{ missing}\}$ versus $\{X > s\}$.
Split B: $\{X \le s\}$ versus $\{X > s \text{ or } X = \text{ missing}\}$.
Split C: $\{X = \text{ missing}\}$ versus $\{X = \text{ not missing}\}$.

Thus, like OTF splitting, one can see that MIA results in a forest ensemble constructed without having to impute data.

2.4 | Unsupervised imputation: RF_{unsv}

RF_{unsv} refers to OTFI using multivariate unsupervised splitting. However unlike the previously desribed OTFI algorithm, the RF_{unsv} algorithm is unsupervised and assumes there is no response (outcome) variable. Instead a multivariate unsupervised splitting rule [10] is implemented. As in the original RF algorithm, at each tree node t, a set of `mtry` variables are selected as potential splitting variables. However, for each of these, as there is no outcome variable, a random set of `ytry` variables is selected and defined to be the multivariate response (pseudo-outcomes). A multivariate composite splitting rule of dimension `ytry` (see below) is applied to each of the `mtry` multivariate regression problems and the node t split on the variable leading to the best split. Missing values in the response are excluded when computing the composite multivariate splitting rule: the split-rule is averaged over non-missing responses only [10]. We also consider an iterated RF_{unsv} algorithm (eg, $\text{RF}_{\text{unsv}}^{(5)}$ implies 5 iterations, RF_{unsv} implies no iterations).

Here is the description of the multivariate composite splitting rule. We begin by considering univariate regression. For notational simplicity, let us suppose the node t we are splitting is the root node based on the full sample size n. Let X be the feature used to split t, where for simplicity we assume X is ordered or numeric. Let s be a proposed split for X that splits t into left and right daughter nodes $t_L := t_L(s)$ and $t_R := t_R(s)$, where $t_L = \{X_i \le s\}$ and $t_R = \{X_i > s\}$. Let $n_L = \#t_L$ and $n_R = \#t_R$ denote the sample sizes for the 2 daughter nodes. If Y_i denotes the outcome, the squared-error split-statistic for the proposed split is

$$D(s,t) = \frac{1}{n}\sum_{i\in t_L}(Y_i - \overline{Y}_{t_L})^2 + \frac{1}{n}\sum_{i\in t_R}(Y_i - \overline{Y}_{t_R})^2, \qquad (1)$$

where $\overline{Y}_{t_L}$ and $\overline{Y}_{t_R}$ are the sample means for t_L and t_R, respectively. The best split for X is the split-point s minimizing $D(s,t)$. To extend the squared-error splitting rule to the multivariate case $q > 1$, we apply univariate splitting to each response coordinate separately. Let $\mathbf{Y}_i = (Y_{i,1}, \ldots, Y_{i,q})^T$ denote the $q \ge 1$ dimensional outcome. For multivariate regression analysis, an averaged standardized variance splitting rule is used. The goal is to minimize

$$D_q(s,t) = \sum_{j=1}^{q}\left\{\sum_{i\in t_L}(Y_{i,j} - \overline{Y}_{t_{L_j}})^2 + \sum_{i\in t_R}(Y_{i,j} - \overline{Y}_{t_{R_j}})^2\right\}, \qquad (2)$$

where $\overline{Y}_{t_{L_j}}$ and $\overline{Y}_{t_{R_j}}$ are the sample means of the jth response coordinate in the left and right daughter nodes. Notice that such a splitting rule can only be effective if each of the coordinates of the outcome are measured on the same scale, otherwise we could have a coordinate j, with say very large values, and its contribution would dominate $D_q(s,t)$. We therefore calibrate $D_q(s,t)$ by assuming that each coordinate has been standardized according to

$$\frac{1}{n}\sum_{i\in t} Y_{i,j} = 0, \quad \frac{1}{n}\sum_{i\in t} Y_{i,j}^2 = 1, \quad 1 \le j \le q. \qquad (3)$$

The standardization is applied prior to splitting a node. To make this standardization clear, we denote the standardized responses by $Y_{i,j}^*$. With some elementary manipulations, it can be verified that minimizing $D_q(s,t)$ is equivalent to maximizing

$$D_q^*(s,t) = \sum_{j=1}^{q}\left\{\frac{1}{n_L}\left(\sum_{i\in t_L} Y_{i,j}^*\right)^2 + \frac{1}{n_R}\left(\sum_{i\in t_R} Y_{i,j}^*\right)^2\right\}. \qquad (4)$$

For multivariate classification, an averaged standardized Gini splitting rule is used. First consider the univariate case (ie,

19321872, 2017, 6, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/sam.11348 by Technical University Dortmund, Wiley Online Library on [22/02/2023]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

the multiclass problem) where the outcome Y_i is a class label from the set $\{1, \ldots, K\}$ where $K \geq 2$. The best split s for X is obtained by maximizing

$$G(s,t) = \sum_{k=1}^{K} \left[\frac{1}{n_L} \left(\sum_{i \in t_L} Z_{i(k)} \right)^2 + \frac{1}{n_R} \left(\sum_{i \in t_R} Z_{i(k)} \right)^2 \right], \quad (5)$$

where $Z_{i(k)} = 1_{\{Y_i = k\}}$. Now consider the multivariate classification scenario $r > 1$, where each outcome coordinate $Y_{i,j}$ for $1 \leq j \leq r$ is a class label from $\{1, \ldots, K_j\}$ for $K_j \geq 2$. We apply Gini splitting to each coordinate yielding the extended Gini splitting rule

$$G_r^*(s,t) = \sum_{j=1}^{r} \left[\frac{1}{K_j} \sum_{k=1}^{K_j} \left\{ \frac{1}{n_L} \left(\sum_{i \in t_L} Z_{i(k),j} \right)^2 + \frac{1}{n_R} \left(\sum_{i \in t_R} Z_{i(k),j} \right)^2 \right\} \right], \quad (6)$$

where $Z_{i(k),j} = 1_{\{Y_{i,j} = k\}}$. Note that the normalization $1/K_j$ employed for a coordinate j is required to standardize the contribution of the Gini split from that coordinate.

Observe that Equations (4) and (6) are equivalent optimization problems, with optimization over $Y_{i,j}^*$ for regression and $Z_{i(k),j}$ for classification. As shown in [9] this leads to similar theoretical splitting properties in regression and classification settings. Given this equivalence, we can combine the 2 splitting rules to form a composite splitting rule. The mixed outcome splitting rule $\Theta(s,t)$ is a composite standardized split rule of mean-squared error Equation (4) and Gini index splitting Equation (6); ie,

$$\Theta(s,t) = D_q^*(s,t) + G_r^*(s,t), \quad (7)$$

where $p = q + r$. The best split for X is the value of s maximizing $\Theta(s,t)$.

Remark 2. As discussed in [19], multivariate regression splitting rules patterned after the Mahalanobis distance can be used to incorporate correlation between response coordinates. Let $\overline{\mathbf{Y}}_L$ and $\overline{\mathbf{Y}}_R$ be the multivariate means for $\mathbf{Y}$ in the left and right daughter nodes, respectively. The following Mahalanobis distance splitting rule was discussed in [19]

$$M_q(s,t) = \sum_{i \in t_L} \left(\mathbf{Y}_i - \overline{\mathbf{Y}}_L \right)^T \hat{\mathbf{V}}_L^{-1} \left(\mathbf{Y}_i - \overline{\mathbf{Y}}_L \right) + \sum_{i \in t_R} \left(\mathbf{Y}_i - \overline{\mathbf{Y}}_R \right)^T \hat{\mathbf{V}}_R^{-1} \left(\mathbf{Y}_i - \overline{\mathbf{Y}}_R \right) \quad (8)$$

where $\hat{\mathbf{V}}_L$ and $\hat{\mathbf{V}}_R$ are the estimated covariance matrices for the left and right daughter nodes. While this is a reasonable approach in low dimensional problems, recall that we are applying $D_q(s,t)$ to `ytry` of the feature variables which could be large if the feature space dimension p is large. Also, because of missing data in the features, it may be difficult to derive estimators for $\hat{\mathbf{V}}_L$ and $\hat{\mathbf{V}}_R$, which is further complicated if their dimensions are high. This problem becomes worse as the tree is grown because the number of observations decreases rapidly making estimation unstable. For these reasons, we use the splitting rule $D_q(s,t)$ rather than $M_q(s,t)$ when implementing imputation.

2.5 | mForest imputation: mRF$_\alpha$ and mRF

The missForest algorithm recasts the missing data problem as a prediction problem. Data are imputed by regressing each variable in turn against all other variables and then predicting missing data for the dependent variable using the fitted forest. With p variables, this means that p forests must be fit for each iteration, which can be slow in certain problems. Therefore, we introduce a computationally faster version of missForest, which we call mForest. The new algorithm is described as follows. Do a quick strawman imputation of the data. The p variables in the data set are randomly assigned into mutually exclusive groups of approximate size αp where $0 < \alpha < 1$. Each group in turn acts as the multivariate response to be regressed on the remaining variables (of approximate size $(1-\alpha)p$). Over the multivariate responses, set imputed values back to missing. Grow a forest using composite multivariate splitting. As in RF$_{\text{unsv}}$, missing values in the response are excluded when using multivariate splitting: the split-rule is averaged over non-missing responses only. Upon completion of the forest, the missing response values are imputed using prediction. Cycle over all of the $\approx 1/\alpha$ multivariate regressions in turn; thus completing 1 iteration. Check if the imputation accuracy of the current imputed data relative to the previously imputed data has increased beyond an ϵ-tolerance value (see Section 3.3 for measuring imputation accuracy). Stop if it has, otherwise repeat until convergence.

To distinguish between mForest under different α values we use the notation mRF$_\alpha$ to indicate mForest fit under the specified α. Note that the limiting case $\alpha = 1/p$ corresponds to missForest. Although technically the 2 algorithms missForest and mRF$_\alpha$ for $\alpha = 1/p$ are slightly different, we did not find significant differences between them during informal experimentation. Therefore for computational speed, missForest was taken to be mRF$_\alpha$ for $\alpha = 1/p$ and denoted simply as mRF.

3 | METHODS

3.1 | Experimental design and data

Table 1 lists the 9 experiments that were carried out. In each experiment, a prespecified target percentage of missing values was induced using 1 of 3 different missing mechanisms [17]:

1. Missing completely at random (MCAR). This means the probability that an observation is missing does not depend on the observed value or the missing ones.
2. Missing at random. This means that the probability of missingness may depend upon observed values, but not missing ones.

19321872, 2017, 6, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/sam.11348 by Technical University Dortmund, Wiley Online Library on [22/02/2023]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

TABLE 1 Experimental design used for large-scale study of RF missing data algorithms

	Missing Mechanism	Percent Missing
EXPT-A	MCAR	25
EXPT-B	MCAR	50
EXPT-C	MCAR	75
EXPT-D	MAR	25
EXPT-E	MAR	50
EXPT-F	MAR	75
EXPT-G	NMAR	25
EXPT-H	NMAR	50
EXPT-I	NMAR	75

3. Not missing at random. This means the probability of missingness depends on both observed and missing values.

Sixty data sets were used, including both real and synthetic data. Figure 1 shows the diversity of the data. Displayed are data sets in terms of correlation (ρ), sample size (n), number of variables (p), and the amount of information contained in the data ($I = \log_{10}(n/p)$). The correlation, ρ, was defined as the L_2-norm of the correlation matrix. If $\mathbf{R} = (\rho_{i,j})$ denotes the $p \times p$ correlation matrix for a data set, ρ was defined to equal

$$\rho = \binom{p}{2}^{-1} \sum_{j=1}^{p} \left(\sum_{k<j} |\rho_{i,j}|^2 \right)^{1/2}. \tag{9}$$

This is similar to the usual definition of the L_2-norm for a matrix, but where we have modifed the definition to remove the diagonal elements of $\mathbf{R}$ which equals 1, as well as the contribution from the symmetric lower diagonal values.

Note that in the plot for p there are 10 data sets with p in the thousands—these are a collection of well-known gene expression data sets. The right-most plot displays the log-information of a data set, $I = \log_{10}(n/p)$. The range of values on the log-scale vary from -2 to 2; thus the information contained in a data set can differ by as much as $\approx 10^4$.

3.2 | Inducing MCAR, MAR, and NMAR missingness

The following procedures were used to induce missingness in the data. Let the target missingness fraction be $0 < \gamma_{\text{NA}} < 1$. For MCAR, data were set to missing randomly without imposing column or row constraints to the data matrix. Specifically, the data matrix was made into a long vector and $n\gamma_{\text{NA}}$ of the entries selected at random and set to missing.

For MAR, missing values were assigned by column. Let $\mathbf{X}_j = (X_{1,j}, \ldots, X_{n,j})$ be the n-dimensional vector containing the original values of the jth variable, $1 \le j \le p$. Each coordinate of $\mathbf{X}_j$ was made missing according to the tail behavior of a randomly selected covariate $\mathbf{X}_k$, where $k \neq j$. The probability of selecting coordinate $X_{i,j}$ was

$$P\{\text{selecting } X_{i,j} | B_j\} \propto \begin{cases} F(X_{i,k}) & \text{if } B_j = 1 \\ 1 - F(X_{i,k}) & \text{if } B_j = 0 \end{cases}, \tag{10}$$

where $F(x) = (1 + \exp(-3x))^{-1}$ and B_j were i.i.d. symmetric 0-1 Bernoulli random variables. With this method, about half of the variables will have higher missingness in those coordinates corresponding to the right tail of a randomly selected variable (the other half will have higher missingness depending on the left tail of a randomly selected variable). A total of $n\gamma_{\text{NA}}$ coordinates were selected from $\mathbf{X}_j$ and set to missing. This induces MAR, as missing values for $\mathbf{X}_j$ depend only on observed values of another variable $\mathbf{X}_k$.

For NMAR, each coordinate of $\mathbf{X}_j$ was made missing according to its own tail behavior. A total of $n\gamma_{\text{NA}}$ values were selected according to

$$P\{\text{selecting } X_{i,j}\} \propto \begin{cases} F(X_{i,j}) & \text{with probability } 1/2 \\ 1 - F(X_{i,j}) & \text{with probability } 1/2. \end{cases} \tag{11}$$

Notice that missingness in $X_{i,j}$ depends on both observed and missing values. In particular, missing values occur with higher probability in the right and left tails of the empirical distribution. Therefore, this induces NMAR.

3.3 | Measuring imputation accuracy

Accuracy of imputation was assessed using the following metric. As described above, values of $\mathbf{X}_j$ were made missing under various missing data assumptions. Let $(1_{1,j}, \ldots, 1_{n,j})$ be a vector of zeroes and ones indicating which values of $\mathbf{X}_j$ were artificially made missing. Define $1_{i,j} = 1$ if $X_{i,j}$ is artificially missing; otherwise $1_{i,j} = 0$. Let $n_j = \sum_{i=1}^{n} 1_{i,j}$ be the number of artificially induced missing values for $\mathbf{X}_j$.

Let $\mathcal{N}$ and $\mathcal{C}$ be the set of nominal (continuous) and categorical (factor) variables with more than 1 artificially induced missing value. That is,

$$\mathcal{N} = \{j : \mathbf{X}_j \text{ is nominal and } n_j > 1\}$$
$$\mathcal{C} = \{j : \mathbf{X}_j \text{ is categorical and } n_j > 1\}.$$

Standardized root-mean-squared error (RMSE) was used to assess performance for nominal variables, and misclassification error for factors. Let $\mathbf{X}_j^*$ be the n-dimensional vector of imputed values for $\mathbf{X}_j$ using procedure $\mathcal{I}$. Imputation error for $\mathcal{I}$ was measured using

$$\mathcal{E}(\mathcal{I}) = \frac{1}{\#\mathcal{N}} \sum_{j \in \mathcal{N}} \sqrt{\frac{\sum_{i=1}^{n} 1_{i,j} \left(X_{i,j}^* - X_{i,j} \right)^2 / n_j}{\sum_{i=1}^{n} 1_{i,j} \left(X_{i,j} - \overline{X}_j \right)^2 / n_j}} + \frac{1}{\#\mathcal{C}} \sum_{j \in \mathcal{C}} \left[\frac{\sum_{i=1}^{n} 1_{i,j} \, 1\{X_{i,j}^* \neq X_{i,j}\}}{n_j} \right],$$

19321872, 2017, 6, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/sam.11348 by Technical University Dortmund, Wiley Online Library on [22/02/2023]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

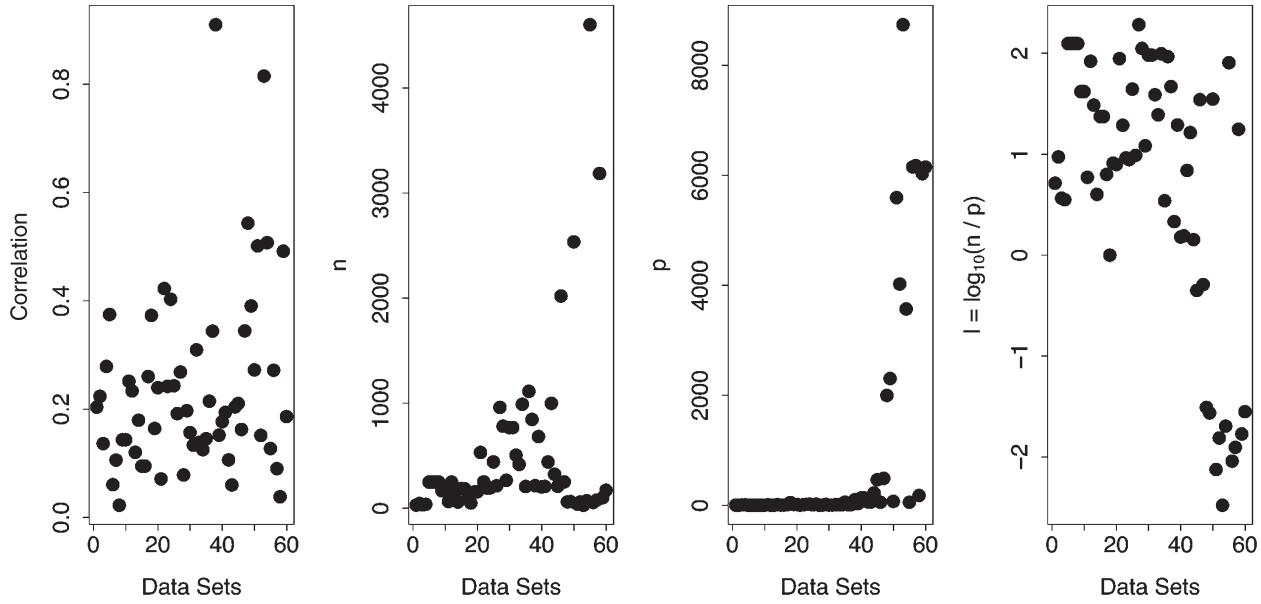

FIGURE 1 Summary values for the 60 data sets used in the large-scale RF missing data experiment. The last panel displays the log-information, $I = \log_{10}(n/p)$, for each data set

where $\overline{X}_j = \sum_{i=1}^{n} \left(1_{i,j} X_{i,j}\right) / n_j$. To be clear regarding the standardized RMSE, observe that the denominator in the first term is the variance of $\mathbf{X}_j$ over the artificially induced missing values, while the numerator is the MSE difference of $\mathbf{X}_j$ and $\mathbf{X}_j^*$ over the induced missing values.

As a benchmark for assessing imputation accuracy we used strawman imputation described earlier, which we denote by $\mathcal{S}$. Imputation error for a procedure $\mathcal{I}$ was compared to $\mathcal{S}$ using relative imputation error defined as

$$\mathcal{E}_R(\mathcal{I}) = 100 \times \frac{\mathcal{E}(\mathcal{I})}{\mathcal{E}(\mathcal{S})}. \tag{12}$$

A value of less than 100 indicates a procedure $\mathcal{I}$ performing better than the strawman.

3.4 | Experimental settings for procedures

Randomized splitting was invoked with an `nsplit` value of 10. For random feature selection, `mtry` was set to $\sqrt{p}$. For random outcome selection for RF_{unsv}, we set `ytry` to equal $\sqrt{p}$. Algorithms RF_{otf} , RF_{unsv} , and RF_{prx} were iterated 5 times in addition to being run for a single iteration. For mForest, the percentage of variables used as responses was $\alpha = .05$ and .25. This implies that $\text{mRF}_{0.05}$ used up to 20 regressions per cycle, while $\text{mRF}_{0.25}$ used 4. Forests for all procedures were grown using a `nodesize` value of 1. Number of trees was set at `ntree` = 500. Each experimental setting (Table 1) was run 100 times independently and results averaged.

For comparison, KNN imputation was applied using the `impute.knn` function from the R-package `impute` [8]. For each data point with missing values, the algorithm determines the KNN using a Euclidean metric, confined to the columns for which that data point is not missing. The missing elements for the data point are then imputed by averaging the non-missing elements of its neighbors. The number of neighbors k was set at the default value $k = 10$. In experimentation we found the method robust to the value of k and therefore opted to use the default setting. Much more important were the parameters `rowmax` and `colmax` which control the maximum percent missing data allowed in each row and column of the data matrix before a rough overall mean is used to impute the row/column. The default values of 0.5 and 0.8, respectively, were too low and led to poor performance in the heavy missing data experiments. Therefore, these values were set to their maximum of 1.0, which greatly improved performance. Our rationale for selecting KNN as a comparison procedure is due to its speed because of the large-scale nature of experiments (total of $100 \times 60 \times 9 = 54\,000$ runs for each method). Another reason was because of its close relationship to forests. This is because RF is also a type of nearest neighbor procedure—although it is an adaptive nearest neighbor. We comment later on how adaptivity may give RF an advantage over KNN.

4 | RESULTS

Section 4.1 presents the results of the performance of a procedure as measured by relative imputation accuracy, $\mathcal{E}_R(\mathcal{I})$. In Section 4.2 we discuss computational speed.

4.1 | Imputation Accuracy

In reporting the values for imputation accuracy, we have stratified data sets into low-, medium-, and high-correlation groups, where correlation, ρ, was defined as in Equation (9). Low-, medium-, and high-correlation groups were defined

19321872, 2017, 6, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/sam.11348 by Technical University Dortmund, Wiley Online Library on [22/02/2023]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

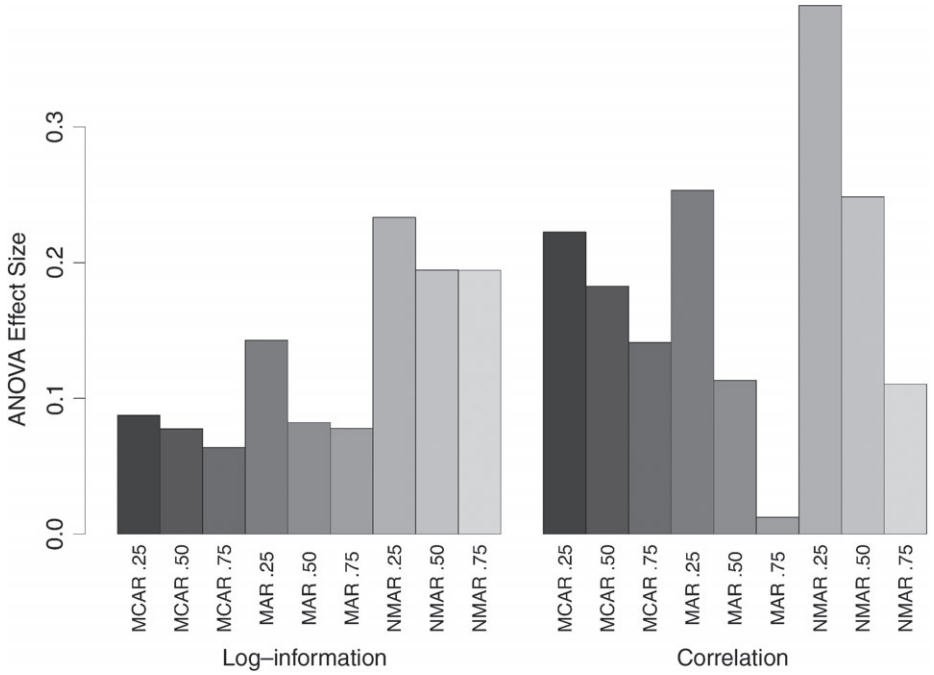

FIGURE 2 ANOVA effect size for the log-information, $I = \log_{10}(n/p)$, and correlation, ρ (defined as in Equation (9)), from a linear regression using log relative imputation error, $\log_{10}(\mathcal{E}_R(\mathcal{I}))$, as the response. In addition to I and ρ, dependent variables in the regression included type of RF procedure used. ANOVA effect sizes are the estimated coefficients of the standardized variable (standardized to have mean 0 and variance 1). This demonstrates the importance of correlation in assessing imputation performance

as groups whose ρ value fell into the $[0, 50]$, $[50, 75]$, and $[75, 100]$, respectively, percentile for correlations. Results were stratified by ρ because we found it played a very heavy role in imputation performance and was much more informative than other quantities measuring information about a data set. Consider for example the log-information for a data set, $I = \log_{10}(n/p)$, which reports the information of a data set by adjusting its sample size by the number of features. While this is a reasonable measure, Figure 2 shows that I is not nearly as effective as ρ in predicting imputation accuracy. The figure displays the ANOVA effect sizes for ρ and I from a linear regression in which log-relative imputation error was used as the response. In addition to ρ and I, dependent variables in the regression also included the type of RF procedure. The effect size was defined as the estimated coefficients for the standardized values of ρ and I. The 2 dependent variables ρ and I were standardized to have a mean of 0 and variance of 1 which makes it possible to directly compare their estimated coefficients. The figure shows that both values are important for understanding imputation accuracy and that both exhibit the same pattern. Within a specific type of missing data mechanism, say MCAR, importance of each variable decreases with missingness of data (MCAR 0.25, MCAR 0.5, MCAR 0.75). However, while the pattern of the 2 measures is similar, the effect size of ρ is generally much larger than I. The only exceptions being the MAR 0.75 and NMAR 0.75 experiments, but these 2 experiments are the least interesting. As will be discussed below, nearly all methods performed poorly here.

4.1.1 | Correlation

Figure 3 and Table 2, which have been stratified by correlation group, show the importance of correlation for RF imputation procedures. In general, imputation accuracy generally improves with correlation. Over the high correlation data, mForest algorithms were by far the best. In some cases, they achieved a relative imputation error of 50, which means their imputation error was half of the strawman's value. Generally there are no noticeable differences between mRF (missForest) and $\text{mRF}_{0.05}$. Performance of $\text{mRF}_{0.25}$, which uses only 4 regressions per cycle (as opposed to p for mRF), is also very good. Other algorithms that performed well in high correlation settings were $\text{RF}^{(5)}_{\text{prxR}}$ (proximity imputation with random splitting, iterated 5 times) and $\text{RF}^{(5)}_{\text{unsv}}$ (unsupervised multivariate splitting, iterated 5 times). Of these, $\text{RF}^{(5)}_{\text{unsv}}$ tended to perform slightly better in the medium- and low-correlation settings. We also note that while mForest also performed well over medium-correlation settings, performance was not superior to other RF procedures in low correlation settings, and sometimes was worse than procedures like $\text{RF}^{(5)}_{\text{unsv}}$. Regarding the comparison procedure KNN, while its performance also improved with increasing correlation, performance in the medium- and low-correlation settings was generally much worse than RF methods.

4.1.2 | Missing data mechanism

The missing data mechanism also plays an important role in accuracy of RF procedures. Accuracy decreased

19321872, 2017, 6, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/sam.11348 by Technical University Dortmund, Wiley Online Library on [22/02/2023]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

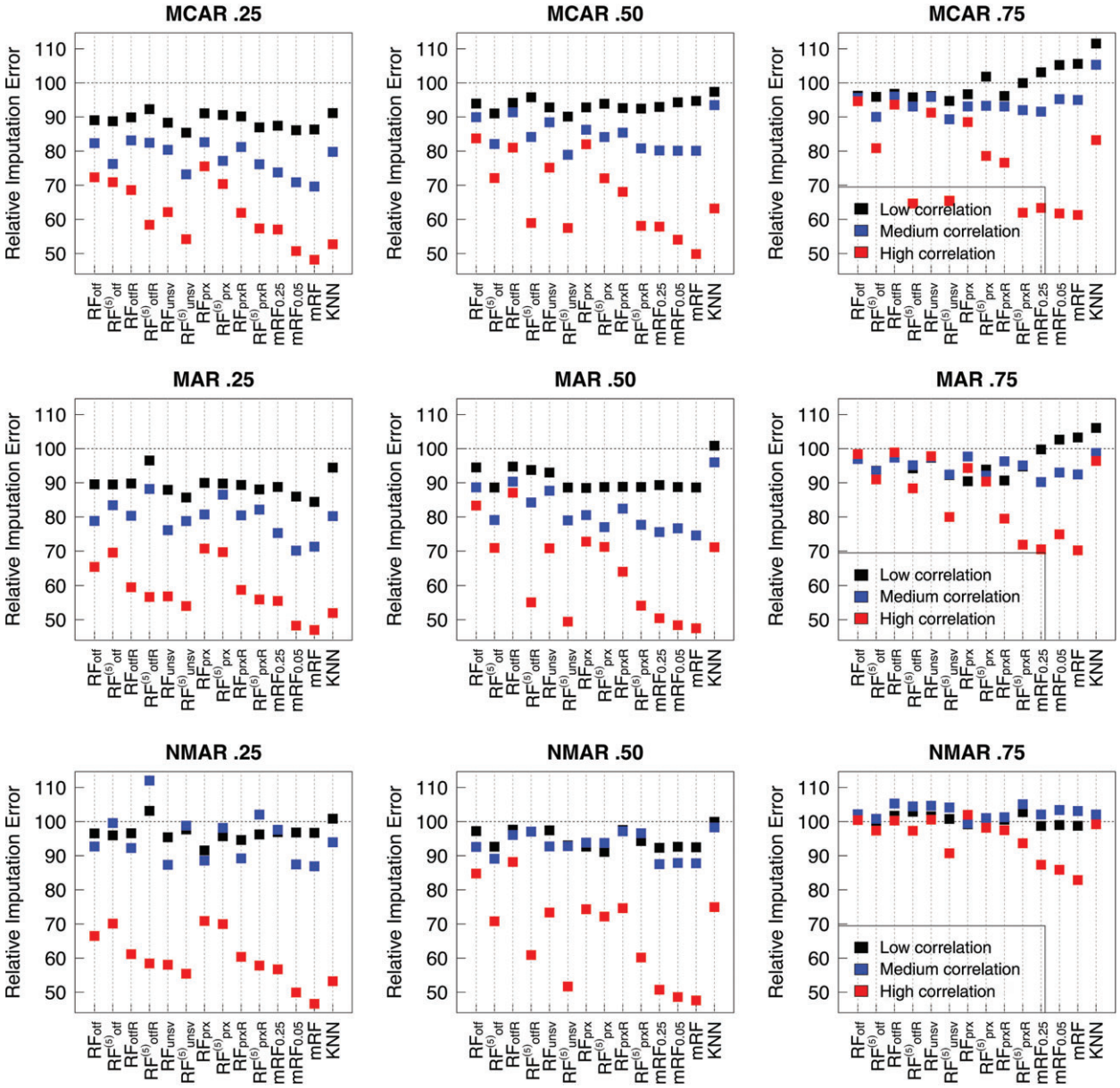

FIGURE 3 Relative imputation error, $\mathcal{E}_R(I)$, stratified and averaged by level of correlation of a data set. Procedures are: RF_{otf}, $\text{RF}^{(5)}_{\text{otf}}$ (on the fly imputation with 1 and 5 iterations); RF_{otfR}, $\text{RF}^{(5)}_{\text{otfR}}$ (similar to RF_{otf} and $\text{RF}^{(5)}_{\text{otf}}$ but using pure random splitting); RF_{unsv}, $\text{RF}^{(5)}_{\text{unsv}}$ (multivariate unsupervised splitting with 1 and 5 iterations); RF_{prx}, $\text{RF}^{(5)}_{\text{prx}}$ (proximity imputation with 1 and 5 iterations); RF_{prxR}, $\text{RF}^{(5)}_{\text{prxR}}$ (same as RF_{prx} and $\text{RF}^{(5)}_{\text{prx}}$ but using pure random splitting); $\text{mRF}_{0.25}$, $\text{mRF}_{0.05}$, mRF (mForest imputation, with 25%, 5% and 1 variable(s) used as the response); KNN (k-nearest neighbor imputation)

systematically when going from MCAR to MAR and NMAR. Except for heavy missingness (75%), all RF procedures under MCAR and MAR were more accurate than strawman imputation. Performance in NMAR was generally poor unless correlation was high.

4.1.3 | Heavy missingness

Accuracy degraded with increasing missingness. This was especially true when missingness was high (75%). For NMAR data with heavy missingness, procedures were not much better than strawman (and sometimes worse), regardless of correlation. However, even with missingness of up to 50%, if correlation was high, RF procedures could still reduce the strawman's error by one-half.

4.1.4 | Iterating RF algorithms

Iterating generally improved accuracy for RF algorithms, except in the case of NMAR data, where in low- and medium-correlation settings, performance sometimes degraded. We believe this is a real effect but have no explanation for this behavior except that it reflects the difficulty in dealing with NMAR.

4.2 | Computational speed

Figure A1 (see Appendix) displays the log of total elapsed time of a procedure averaged over all experimental conditions and runs, with results ordered by the log-computational complexity of a data set, $c = \log_{10}(np)$. The fastest algorithm is

19321872, 2017, 6, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/sam.11348 by Technical University Dortmund, Wiley Online Library on [22/02/2023]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

TABLE 2 Relative imputation error $\mathcal{E}_R(I)$

	Low correlation								
	MCAR			MAR			NMAR		
	.25	.50	.75	.25	.50	.75	.25	.50	.75
RF_{otf}	89.0	93.9	96.2	89.5	94.5	97.2	96.5	97.2	100.9
$RF^{(5)}_{otf}$	88.7	91.0	95.9	89.5	88.6	93.5	96.0	92.6	98.8
RF_{otfR}	89.9	94.1	96.8	89.8	94.7	97.8	96.6	97.6	101.7
$RF^{(5)}_{otfR}$	92.3	95.8	95.8	96.5	93.7	94.2	103.2	97.0	102.9
RF_{unsv}	88.3	92.8	96.2	87.9	93.0	97.3	95.4	97.4	101.6
$RF^{(5)}_{unsv}$	85.4	90.1	94.7	85.7	88.6	92.2	97.7	93.0	100.8
RF_{prx}	91.1	92.8	96.7	89.9	88.5	90.4	91.5	92.7	99.2
$RF^{(5)}_{prx}$	90.6	93.9	101.8	89.8	88.7	93.9	95.7	91.1	99.3
RF_{prxR}	90.2	92.6	96.2	89.4	88.8	90.7	94.6	97.5	100.5
$RF^{(5)}_{prxR}$	86.9	92.4	100.0	88.1	88.8	94.8	96.2	94.3	102.8
$mRF_{0.25}$	87.4	92.9	103.1	88.8	89.3	99.8	96.9	92.3	98.7
$mRF_{0.05}$	86.1	94.3	105.3	86.0	88.7	102.7	96.8	92.6	99.0
mRF	86.3	94.7	105.6	84.4	88.6	103.3	96.7	92.5	98.8
KNN	91.1	97.4	111.5	94.4	100.9	106.1	100.9	100.0	101.7

	Medium correlation								
	MCAR			MAR			NMAR		
	.25	.50	.75	.25	.50	.75	.25	.50	.75
RF_{otf}	82.3	89.9	95.6	78.8	88.6	97.0	92.7	92.6	102.2
$RF^{(5)}_{otf}$	76.2	82.1	90.0	83.4	79.1	93.4	99.6	89.1	100.8
RF_{otfR}	83.1	91.4	96.0	80.3	90.3	97.4	92.2	96.1	105.3
$RF^{(5)}_{otfR}$	82.4	84.1	93.1	88.2	84.2	95.1	112.0	97.1	104.5
RF_{unsv}	80.4	88.4	95.9	76.1	87.7	97.5	87.3	92.7	104.7
$RF^{(5)}_{unsv}$	73.2	78.9	89.3	78.8	79.0	92.4	98.8	92.8	104.2
RF_{prx}	82.6	86.3	93.1	80.7	80.5	97.7	88.6	93.8	99.5
$RF^{(5)}_{prx}$	77.1	84.1	93.3	86.5	77.0	92.1	98.1	93.7	101.0
RF_{prxR}	81.2	85.4	93.1	80.4	82.4	96.3	89.2	97.2	101.3
$RF^{(5)}_{prxR}$	76.1	80.8	92.0	82.1	77.7	95.1	102.1	96.6	105.1
$mRF_{0.25}$	73.8	80.2	91.6	75.3	75.6	90.2	97.6	87.5	102.1
$mRF_{0.05}$	70.9	80.1	95.2	70.1	76.6	93.0	87.4	87.9	103.4
mRF	69.6	80.1	95.0	71.3	74.6	92.4	86.9	87.8	103.1
KNN	79.8	93.5	105.3	80.2	96.0	98.7	93.9	98.3	102.1

	High correlation								
	MCAR			MAR			NMAR		
	.25	.50	.75	.25	.50	.75	.25	.50	.75
RF_{otf}	72.3	83.7	94.6	65.5	83.3	98.4	66.5	84.8	100.4
$RF^{(5)}_{otf}$	70.9	72.1	80.9	69.5	70.9	91.0	70.1	70.8	97.3
RF_{otfR}	68.6	81.0	93.6	59.5	87.1	98.9	61.2	88.2	100.3
$RF^{(5)}_{otfR}$	58.4	58.9	64.6	56.7	55.1	88.4	58.4	60.9	97.3
RF_{unsv}	62.1	75.1	91.3	56.8	70.8	97.8	58.1	73.3	100.6
$RF^{(5)}_{unsv}$	54.2	57.5	65.4	54.0	49.4	80.0	55.4	51.7	90.7
RF_{prx}	75.5	82.0	88.5	70.7	72.8	94.3	70.9	74.3	102.0
$RF^{(5)}_{prx}$	70.4	72.0	78.6	69.7	71.2	90.3	70.0	72.2	98.2
RF_{prxR}	61.9	68.1	76.6	58.7	64.1	79.5	60.4	74.6	97.5
$RF^{(5)}_{prxR}$	57.3	58.1	61.9	55.9	54.1	71.9	57.8	60.2	93.7
$mRF_{0.25}$	57.0	57.9	63.3	55.5	50.4	70.5	56.7	50.7	87.3
$mRF_{0.05}$	50.7	54.0	61.7	48.3	48.4	74.9	49.9	48.6	85.9
mRF	48.2	49.8	61.3	47.0	47.5	70.2	46.6	47.6	82.9
KNN	52.7	63.2	83.2	52.0	71.1	96.4	53.2	74.9	99.2

19321872, 2017, 6, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/sam.11348 by Technical University Dortmund, Wiley Online Library on [22/02/2023]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

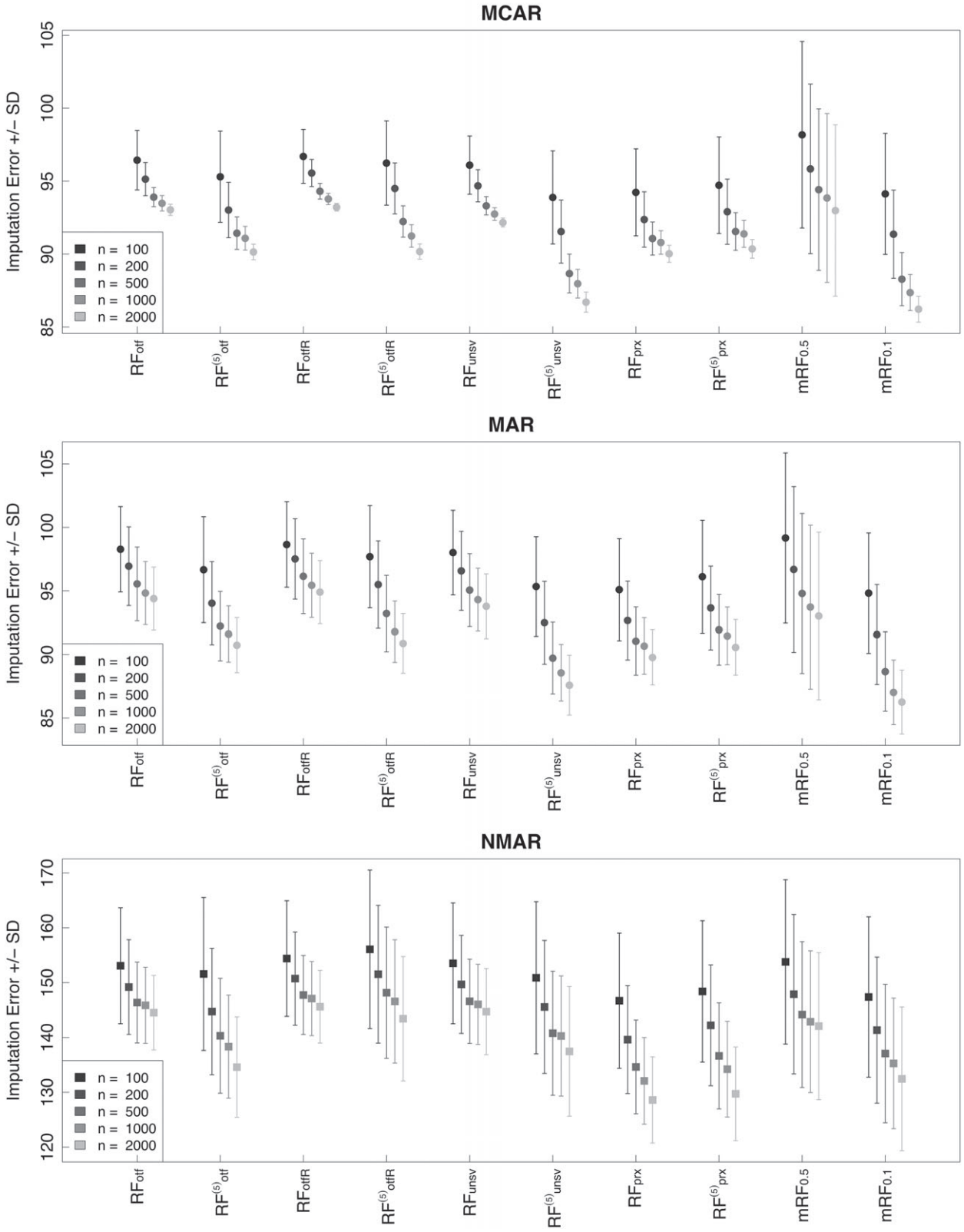

FIGURE 4 Mean relative imputation error ± SD from simulations under different sample size values $n = 100, 200, 500, 1000, 2000$

KNN which is generally 3 times faster on the log-scale, or 1000 times faster, than the slowest algorithm, mRF (missForest). To improve clarity of these differences, Figure A2 (see Appendix) displays the relative computational time of procedure relative to KNN (obtained by subtracting the KNN log-time from each procedure's log-time). This new figure shows that while mRF is 1000 times slower than KNN, the multivariate mForest algorithms, $\text{mRF}_{0.05}$ and $\text{mRF}_{0.25}$, improve speeds by about a factor of 10. After this, the next slowest procedures are the iterated algorithms. Following this

19321872, 2017, 6, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/sam.11348 by Technical University Dortmund, Wiley Online Library on [22/02/2023]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

TABLE 3 Relative imputation error $\mathcal{E}_R(\mathcal{I})$ with and without a pattern-mixture model (PMM) for Y. Some procedures, such as mForest, do not use Y outcome in imputations. Therefore, their imputation performance is the same with or without PMM. For clarity, we therefore only report values without PMM for such procedures

	MCAR	MCAR (PMM)	MAR	MAR (PMM)	NMAR	NMAR (PMM)
RF_{otf}	91.8	92.5	91.1	91.9	94.5	95.6
$RF_{otf}^{(5)}$	88.9	90.1	87.6	88.7	88.0	91.5
RF_{prx}	88.8	90.1	86.7	87.5	84.8	88.0
$RF_{prx}^{(5)}$	89.1	90.1	87.4	88.1	84.8	89.1
RF_{unsv}	90.9	–	90.5	–	94.6	–
$RF_{unsv}^{(5)}$	85.1	–	84.5	–	90.0	–
RF_{prxR}	88.3	–	86.1	–	92.3	–
$RF_{prxR}^{(5)}$	88.4	–	86.4	–	93.3	–
$mRF_{0.5}$	91.7	–	89.8	–	92.9	–
$mRF_{0.1}$	85.1	–	83.3	–	86.6	–

are the non-iterated algorithms. Some of these latter algorithms, such as RF_{otf}, are 100 times faster than missForest; or only 10 times slower than KNN. These kinds of differences can have a real effect when dealing with big data. We have experienced settings where OTF algorithms can take hours to run. This means that the same data would take missForest hundreds of hours to run, which makes it questionable to be used in such settings.

5 | SIMULATIONS

5.1 | Sample size

We used simulations to study the performance of RF as the sample size n was varied. We wanted to investigate 2 questions: (1) Does the relative imputation error improve with sample size? (2) Do these values converge to the same or different values for the different RF imputation algorithms?

For our simulations, there were 10 variables $X_1, \ldots, X_{10}$ where the true model was

$$Y = X_1 + X_2 + X_3 + X_4 + \varepsilon, \tag{13}$$

where ε was simulated independently from a $N(0, 0.5)$ distribution (here $N(\mu, v)$ denotes a normal distribution with mean μ and variance v). Variables X_1 and X_2 were correlated with a correlation coefficient of 0.96, and X_5 and X_6 were correlated with value 0.96. The remaining variables were not correlated. Variables X_1, X_2, X_5, X_6 were $N(3, 3)$, variables X_3, X_{10} were $N(1, 1)$, variable X_8 was $N(3, 4)$, and variables X_4, X_7, X_9 were exponentially distributed with mean 0.5.

The sample size (n) was chosen to be 100, 200, 500, 1000, and 2000. Data were made missing using the MCAR, MAR, and NMAR missing data procedures described earlier. Percentage of missing data was set at 25%. All imputation parameters were set to the same values as used in our previous experiments as described in Section 3.4. Each experiment was repeated 500 times and the relative imputation error, $\mathcal{E}_R(\mathcal{I})$, recorded in each instance. Figure 4 displays the mean relative imputation error for a RF procedure and its standard deviation for each sample size setting. As can be seen, values improve with increasing n. It is also noticeable that performance depends upon the RF imputation method. In these simulations, the missForest algorithm $mRF_{0.1}$ appears to be best (note that $p = 10$ so $mRF_{0.1}$ corresponds to the limiting case missForest). Also, it should be noted that performance of RF procedures decrease systematically as the missing data mechanism becomes more complex. This mirrors our previous findings.

5.2 | Pattern mixture models

In pattern-mixture models [14,15], the outcome is changed when an independent variable is missing. This is considered a challenging missing data scenario, therefore we sought to investigate performance of procedures in this setting. We used simulations to compare imputation performance of RF methods when the missing data mechanism was MCAR, MAR, and NMAR for the X features, and where the outcome Y was simulated with and without a pattern-mixture model (PMM). Independent variables $X_1, \ldots, X_{10}$ were simulated as in Section 5.1. The PMM was defined as

$$Y = X_1 + X_2 + X_3 + X_5 + 10M_1 + \varepsilon, \tag{14}$$

where ε was simulated independently from a $N(0, 0.5)$ distribution. Pattern-mixture missingness was induced by M_1 which was set to $M_1 = 1$ if X_1 was missing and $M_1 = 0$ if X_1 was not missing. This results in the value of Y being affected by the missingness of X_1. Missingness for $X_1, \ldots, X_{10}$ was induced by MCAR, MAR, and NMAR as in Section 5.1. The sample size was chosen to be $n = 2000$. Simulations were repeated 500 times.

The results are displayed in Table 3. Relative imputation error for each procedure is given for each of the 3 X missing data mechanisms with and without a PMM for Y. Because methods such as mForest do not use Y in their imputation procedure, only the results from simulations without PMM are displayed for such procedures. As shown in Table 3, $mRF_{0.1}$ generally had the best imputation accuracy (as before,

19321872, 2017, 6, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/sam.11348 by Technical University Dortmund, Wiley Online Library on [22/02/2023]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

note that $mRF_{0.1}$ corresponds to the limiting case missForest because $p = 10$). For PMM simulations, OTF and proximity methods, which use Y in their model building, generally saw a degradation in imputation performance, especially when missingness for X was NMAR. This shows the potential danger of imputation procedures which include Y, especially when missingness in X has a complex relationship to Y. A more detailed study of this issue will be undertaken by the authors in a follow-up paper.

6 | CONCLUSIONS

Being able to effectively impute missing data is of great importance to scientists working with real world data today. A machine learning method such as RF, known for its excellent prediction performance and ability to handle all forms of data, represents a poentially attractive solution to this challenging problem. However, because no systematic comparative study of RF had been attempted in missing data settings, we undertook a large-scale experimental study of different RF imputation procedures to determine which methods performed best, and under what types of settings.

We found that correlation played a very strong role in performance of RF procedures. Imputation performance generally improved with increasing correlation of features. This held even with heavy levels of missing data and for all but the most complex missing data scenarios. When there is high correlation we recommend using a method like missForest which performed the best in such settings. Although it might seem obvious that increasing feature correlation should improve imputation, we found that in low to medium correlation, RF algorithms did noticeably better than the popular KNN imputation method. This is interesting because KNN is related to RF. Both methods are a type of nearest neighbor method, although RF is more adaptive than KNN, and in fact can be more accurately described as an adaptive nearest neighbor method. This adaptivity of RF may play a special role in harnessing correlation in the data that may not necessarily be present in other methods, even methods that have similarity to RF. Thus, we feel it is worth emphasizing that correlation is extremely important to RF imputation methods.

In big data settings, computational speed will play a key role. Thus, practically speaking, users might not be able to implement the best method because computational times will simply be too long. This is the downside of a method like missForest, which was the slowest of all the procedures considered. As a solution, we proposed mForest (mRF_{α}) which is a computationally more efficient implementation of missForest. Our results showed mForest could achieve up to a 10-fold reduction in compute time relative to missForest. We believe these computational times can be improved further by incorporating mForest directly into the native C-library of RF-SRC. Currently mForest is run as an external R-loop that makes repeated calls to the `impute.rfsrc` function in RF-SRC. Incorporating mForest into the native library, combined with the openMP parallel processing of RF-SRC, could make it much more attractive. However, even with all of this, we still recommend some of the more basic OTFI algorithms like unsupervised RF imputation procedures for big data. These algorithms perform solidly in terms of imputation and can be up to a 100 times faster than missForest.

ACKNOWLEDGMENTS

This work was supported by the National Institutes of Health [R01CA163739 to H.I.].

REFERENCES

1. T. Aittokallio, *Dealing with missing values in large-scale studies: Microarray data imputation and beyond*, Brief Bioinform. **2** (2009), no. 2, 253–264.
2. J. W. Bartlett et al., *Multiple imputation of covariates by fully conditional specification: accommodating the substantive model*, Stat. Methods Med. Res. **24** (2015), no. 4, 462–487.
3. L. Breiman, *Random forests*, Mach. Learn. **45** (2001), 5–32.
4. L. Breiman, *Manual–Setting up, using, and understanding random forests V4.0*, 2003. Available at https://www.stat.berkeley.edu/ breiman.
5. L. Breiman et al., *Classification and regression trees*, Belmont, California (1984).
6. L. L. Doove, S. Van Buuren, and E. Dusseldorp, *Recursive partitioning for missing data imputation in the presence of interaction effects*, Comput. Stat. Data Anal. **72** (2014), 92–104.
7. C. K. Enders, *Applied missing data analysis* Guilford Publications, New York, 2010.
8. T. Hastie et al., 2015. http://bioconductor.org.
9. H. Ishwaran, *The effect of splitting on random forests*, Mach. Learn. **99** (2015), no. 1, 75–118.
10. H. Ishwaran and U. B. Kogalur, *randomForestSRC: Random forests for survival, regression and classification (RF-SRC)*, 2017. R package version 2.4.2 http://cran. r-project.org.
11. H. Ishwaran et al., *Random survival forests*, Ann. Appl. Stat. **2** (2008), 841–860.
12. Liao S. G. et al., *Missing value imputation in high-dimensional phenomic data: Imputable or not, and how?*, BMC Bioinform. **15** (2014), 346.
13. A. Liaw and M. Wiener, *Classification and regression by randomForest*, Rnews **2** (2002), no. 3, 18–22.
14. R. J. A. Little, *Pattern-mixture models for multivariate incomplete data*, J Am. Stat. Assoc. **88** (1993), no. 421, 125–134.
15. R. J. A. Little and D. B. Rubin John Wiley & Sons, 2014.
16. P. L. Loh and M. J. Wainwright, *High-dimensional regression with noisy and missing data: provable guarantees with non-convexity*, Adv. Neural Inf. Process. Syst. (2011), 2726–2734.
17. D. B. Rubin, *Inference and missing data*, Biometrika **63** (1976), no. 3, 581–592.
18. D. B. Rubin, *Multiple imputation after 18+ years*, J. Am. Stat. Assoc. **91** (1996), 473–489.
19. M. Segal and Y. Xiao, *Multivariate random forests*, WIREs: Data Min. Knowl. Discov. **1** (2011), no. 1, 80–87.
20. A. D. Shah et al., *Comparison of random forest and parametric imputation models for imputing missing data using MICE: A CALIBER study*, Am. J. Epidemiol. **179** (2014), no. 6, 764–774.
21. D. J. Stekhoven and P. Buhlmann, *MissForest—Non-parametric missing value imputation for mixed-type data*, Bioinformatics **28** (2012), no. 1, 112–118.
22. O. Troyanskaya et al., *Missing value estimation methods for DNA microarrays*, Bioinformatics **17** (2001), no. 6, 520–525.

19321872, 2017, 6, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/sam.11348 by Technical University Dortmund, Wiley Online Library on [22/02/2023]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

23. B. Twala and M. Cartwright, *Ensemble missing data techniques for software effort prediction*, Intell. Data Anal. **14** (2010), no. 3, 299–331.
24. B. Twala, M. C. Jones, and D. J. Hand, *Good methods for coping with missing data in decision trees*, Pattern Recognit. Lett. **29** (2008), no. 7, 950–956.
25. S. Van Buuren, *Multiple imputation of discrete and continuous data by fully conditional specification*, Stat. Methods Med. Res. **16** (2007), 219–242.
26. A. K. Waljee et al., *Comparison of imputation methods for missing laboratory data in medicine*, BMJ Open **3** (2013), no. 8, e002847.

How to cite this article: Tang F, Ishwaran H. Random Forest Missing Data Algorithms. *Stat Anal Data Min: The ASA Data Sci Journal*. 2017;10:363–377. https://doi.org/10.1002/sam.11348.

APPENDIX

Figures A1 and A2 display computational speed for different RF algorithms as a function of complexity.

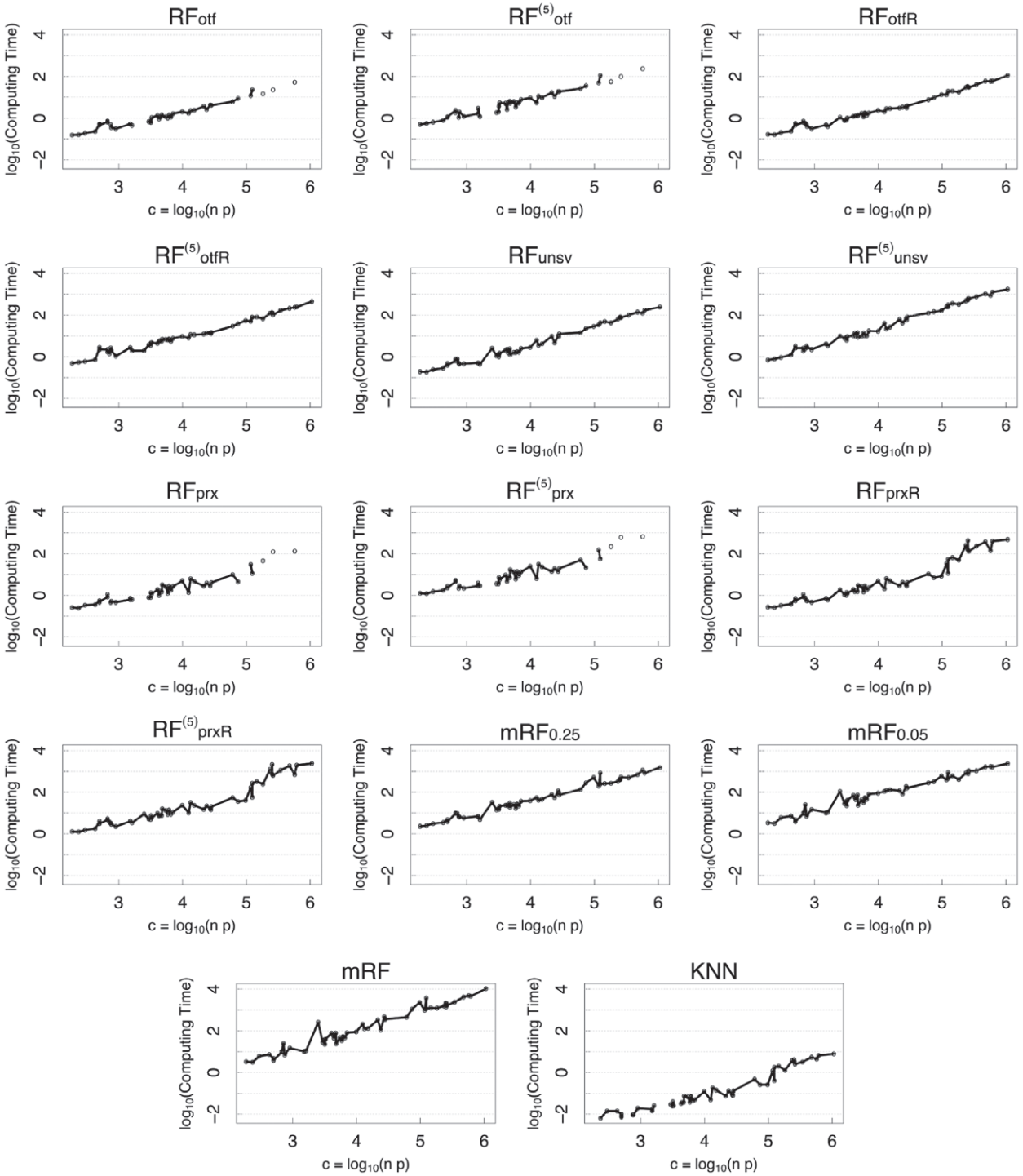

FIGURE A1 Log of computing time for a procedure versus log-computational complexity of a data set, $c = \log_{10}(np)$

19321872, 2017, 6, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/sam.11348 by Technical University Dortmund, Wiley Online Library on [22/02/2023]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

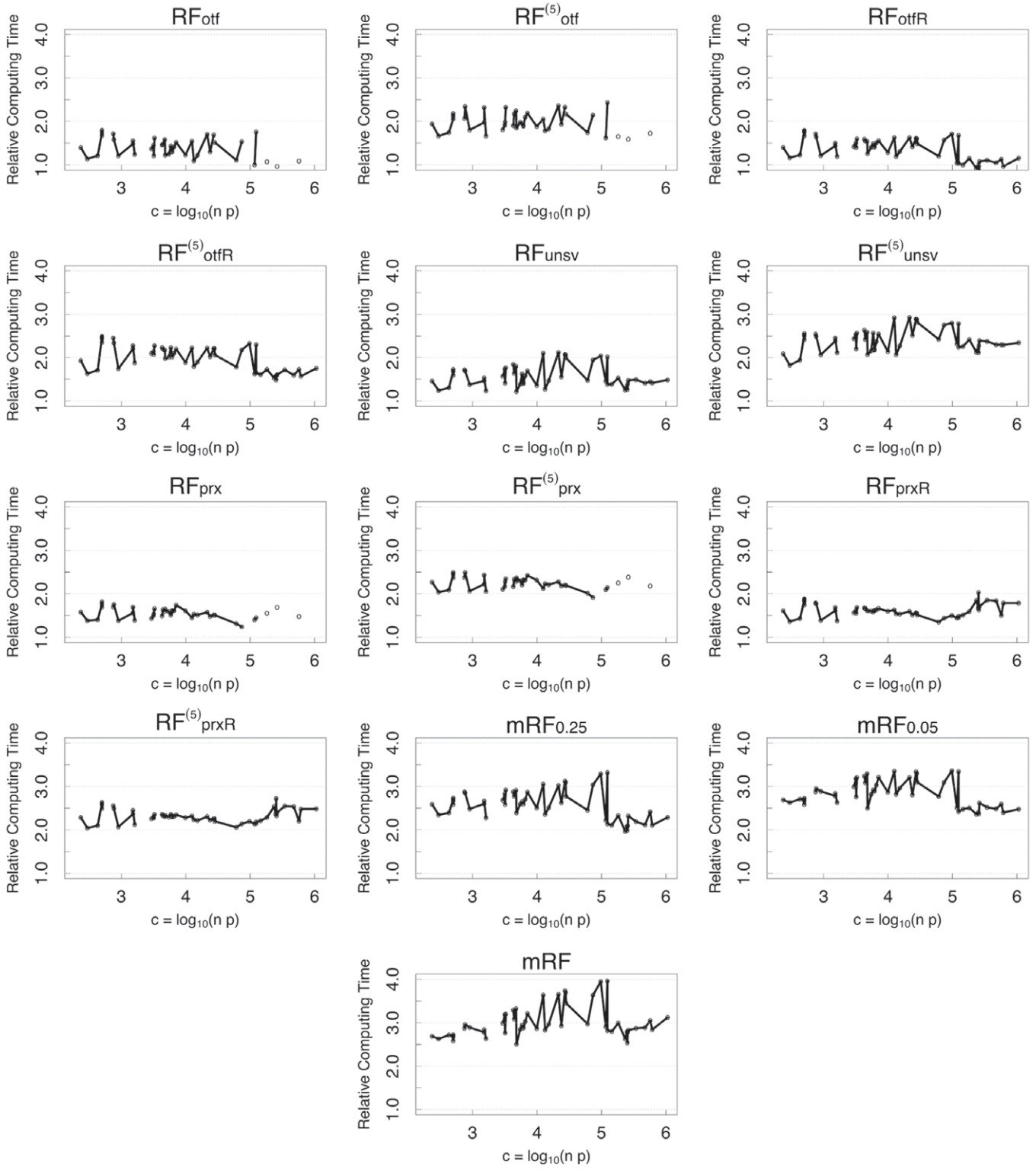

FIGURE A2 Relative log-computing time (relative to KNN) versus log-computational complexity of a data set, $c = \log_{10}(np)$

19321872, 2017, 6, Downloaded from https://onlinelibrary.wiley.com/doi/10.1002/sam.11348 by Technical University Dortmund, Wiley Online Library on [22/02/2023]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License